A TECHNIQUE FOR SPEEDING CONVERGENCE IN SOLVING LINEAR PROGRAMS

JOHN EDMUND EASTERBROOK

A TECHNIQUE FOR SPEEDING CONVERGENCE IN SOLVING LINEAR PROGRAMS

by

John Edmund Easterbrook
Captain, United States Army
B.S., United States Military Academy, 1962

[redacted]

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL
June 1968

ABSTRACT

A technique for reducing the number of iterations necessary for solving linear programs using the primal-dual algorithm is presented. It appears that the new method will also decrease the number of iterations over any other simplex algorithm. A FORTRAN program incorporating the technique, as well as some comparative computational results are given.

TABLE OF CONTENTS

LIST OF TABLES

SECTION I

INTRODUCTION

The primal-dual algorithm, as developed by Ford and Fulkerson, theoretically permitted faster convergence in solving linear programs over such methods as the two-phase or Charnes' M-method. Instead of first driving the artificial variables to zero to obtain a feasible solution and then working to optimality in a second phase, the primal-dual works towards optimality and feasibility in the primal and dual problems at the same time. This is accomplished during what would be the first phase for the other procedures. However, in actual computational experience, the primal-dual algorithm has not achieved faster convergence over the other simplex procedures.

This paper presents[1] a modification of the primal-dual algorithm that does accomplish faster convergence than the usual primal-dual algorithm and perhaps also the other simplex methods.

[1] Professor Harold Greenberg provided the idea for the new technique.

SECTION II

THE STANDARD PRIMAL-DUAL ALGORITHM

Using matrix notation, the linear programming problem is:

$$\text{Minimize} \quad z = z_o + CX$$

$$\text{Given} \quad AX = H$$

$$x_j \geq 0 \qquad j = 1,\ldots,n$$

where z_o is a constant and

C is a 1 x n vector with components $c_j \geq 0$, $j = 1,\ldots,n$

H is an m x 1 vector with components $h_i \geq 0$, $i = 1,\ldots,m$

A is an m x n matrix with elements $a_{i,j}$, $i = 1,\ldots,m$
$j = 1,\ldots,n$

X is an n x 1 vector with components x_j, $j = 1,\ldots,n$

The standard primal-dual, as it appears in Dantzig,[2] consists in augmenting the constraint equations with artificial variables as,

$$AX + Y = H \tag{1}$$

$$y_i \geq 0 \qquad i = 1,\ldots,m$$

where Y is an m x 1 vector with components consisting of the artificial variables y_i; and forming

$$w = w_o + DX$$

where

$$w_o = \sum_{i=1}^{m} h_i$$

and D is a 1 x n vector with components d_j, $j = 1,\ldots,n$ where

[2]George B. Dantzig, Linear Programming and Extensions (Princeton: Princeton University Press, 1963), pp. 247-53.

$$d_j = - \sum_{i=1}^{m} a_{i,j}$$

It is noted that both the c_j and the h_i must be positive prior to starting the computations. Negative h_i may be made positive by multiplying the associated constraint equations by a minus one. The method for correcting negative c_j is discussed in the "Description of the Computer Program" section of this paper.

The problem is to find $x_j \geq 0$, $w = 0$, that minimizes z subject to (1). At any stage of the iterative procedure the equations have the form:

$$\begin{aligned} X_B + \overline{A}_1 X &= \overline{H}_1 \\ Y + \overline{A}_2 X &= \overline{H}_2 \\ \overline{D}X &= w - \overline{w}_o \\ \overline{C}X &= z - \overline{z}_o \end{aligned} \tag{2}$$

where X_B is a vector representing the current basic variables and X is a vector representing the current non-basic variables. The numerical subscripts 1 and 2 represent partitioned matrices and vectors. Matrices and vectors with bars are the corresponding original matrices and vectors after the usual pivot operations within the primal and dual problems. When an artificial variable becomes non-basic, it is dropped from further consideration. This results in the $\overline{A}_1$ matrix and the $\overline{H}_1$ vector. Each iteration in the standard primal-dual algorithm consists of the following:

a) Finding the most negative $\overline{d}_j$ value for those indices j having $\overline{c}_j = 0$.

b) If there are no indices with this property, the multiplier $k = \underset{\bar{d}_j < 0}{\min} [\bar{c}_j / -\bar{d}_j]$ is used to obtain at least one $\bar{c}_j$ equal to zero by

$$\bar{c}_j = \bar{c}_j + k(d_j)$$

c) The index $r = j$ is obtained for the minimum $\bar{d}_j$ with $\bar{c}_j = 0$. Then x_r is selected to be basic.

d) Then min $(\bar{h}_i / a_{i,r})$ is found for $a_{i,r} > 0$. This will select a pivot row in either the X_B or Y sets of equations.

e) The simplex pivot method is then used. If the pivot row is in the Y set of equations the artificial variable in that row is dropped. Thus, in any case, one of the forms in (2) is achieved.

When $\bar{w}_o = 0$, the basic solution is optimal and the problem is solved.

The actual computations should include rules to prevent cycling, such as the perturbation technique or the use of lexicographic ordering.

SECTION III

DESCRIPTION OF THE NEW TECHNIQUE

The primal-dual algorithm is modified in a two-step procedure. In the first step, a pivot row in the Y set of equations is selected. By only considering elimination of artificial variables, after m or less iterations, w equals zero. At this point the equations have the form:

$$X_B + \bar{A}X = \bar{H}$$
$$z_o + \bar{C}X = \bar{z}$$

where all $\bar{c}_j$ are positive. If all the $\bar{h}_i$ are positive, the solution is optimal. We then stop. However, in most cases, this stage of the problem will contain one or more negative $\bar{h}_i$. In this case the subscript "it" is assigned to the row with the most negative $\bar{h}_i$. The following procedure is then used as the second step of the modification:

a) The equation containing $\bar{h}_{it}$ is multiplied through by minus one. This produces $\bar{h}_{it} > 0$.

b) The "it" equation is then added to all equations with negative $\bar{h}_i$. All basic variables except the "it" equation now become feasible.

c) To maintain the canonical form an artificial variable is added to the "it" equation. A new infeasibility equation with $\bar{w}_o = \bar{h}_{it}$ is formed from the "it" equation.

The process now reverts to the primal-dual algorithm to achieve $\bar{w}$ equal to zero. Since the canonical form has been maintained and all $\bar{c}_j$ have been kept positive throughout the process, once $\bar{w}$ equals zero, the solution is optimal.

SECTION IV

COMPUTATIONAL RESULTS

Several problems were run on an IBM 360/67 using both the standard primal-dual and the primal-dual with the new technique. The problems used were test problems from the SHARE Linear Programming Project and two highly degenerate assignment type problems. Results are given in Table I with the two assignment problems listed first.

TABLE I

TABLE OF COMPARATIVE RESULTS

PROBLEM	SIZE (m x n)	SIMPLEX ITERATIONS[3]	PRIMAL-DUAL WITH THE NEW TECHNIQUE Iterations	Seconds	STANDARD PRIMAL-DUAL Iterations	Seconds
Assignment	74 x 150		89	32	83	29
Assignment	34 x 376		74	12	97	28
SHARE 1A	33 x 64	70	39	7	39	7
SHARE 1D	27 x 45	61	38		38	
SHARE 1E	31 x 106	66	96	30	136	42
SHARE 1F	66 x 135	184	125	86	173	114
SHARE 2A	30 x 103	114	79	29	119	43

In the two assignment problems the primal-dual with the new technique resulted in optima different from that of the standard primal-dual since

[3]Philip Wolfe and Leola Cutler, "Experiments in Linear Programming," <u>Recent Advances in Mathematical Programming</u> (New York: McGraw-Hill Book Co., Inc., 1963), p. 198.

these problems have non-unique solutions. The two methods resulted in the same optima in all other problems.

No accurate timing data is available. The timing data in the above table though does give an indication of the reduction in execution times possible with the new technique. However, the modified method does perform less work than the usual method because in the first step ratios and comparisons are only taken for the artificial rows. The restart of the problem in the second step requires only an insignificant amount of time. Thus, in the table, the number of iterations represents a measure of improvement of the new method. The new technique shows significant reductions in the number of iterations from the two-phase standard simplex method in four out of the five problems where a comparison is possible. When comparing the new technique with the standard primal-dual, a similar reduction is evident in the larger problems. It therefore appears that the new method presents a significant improvement when the number of variables is large.

SECTION V

DESCRIPTION OF THE COMPUTER PROGRAM

The accompanying program (see Appendix) is written in the FORTRAN IV language, and was used to run all test problems with the exception of the two assignment type problems (since these only deal with zeros or ones in the A matrix, considerable reduction in the core memory requirement could be made with a slight modification of the program). The program will handle up to 69 constraint equations and up to 149 variables, and requires about 160,000 bits of core storage. Columns may be designated by alphanumeric names, but rows must be consecutive integer values not to exceed the value 69. Only the non-zero elements of A, C, and H need be read in. All h_i must be positive, but negative c_j are allowed.

Names used in this program conform to those already outlined in this paper. Those not previously discussed are defined as:

FF is a multiplicative constant used to insure that all $a_{i,j}$ are integer.

EP is a constant used in the round-off error control.

DI is the cumulative multiplication of the pivot elements.

CE is a vector identical to the initial C, and is used to calculate z.

AX is the pivot column in the Revised Simplex tableau.

CNAME is a vector of column names.

DUAL is the vector of dual variables.

DB is a vector of simplex multipliers used to calculate the d_j.

R is a vector used to record whether each variable is basic or artificial.

E is a computed scalar used in the updating and round-off control.

P is a vector of multipliers used in the actual pivot operation.

ITR indicates the number of iterations performed.

INDEX controls the use of the new technique (INDEX equals zero yields the standard primal-dual, and INDEX equals one incorporates the new technique).

IB is a vector of variable designators (a negative number indicates an artificial variable, and a positive number indicates a basic variable).

All other names are defined in the program.

Once the data has been read in, the c_j are tested for negative values. If all c_j are positive, the primal-dual commences. However, if there is one or more negative c_j the following procedure is followed:

a) RMIN is set equal to the minimum c_j.

b) n is set equal to n + 1 and m is set equal to m + 1.

c) The coefficients of all variables in the added constraint equation indicated in b) above are set equal to FF including x_n. (Variable x_n does not appear in any other constraint equation.)

d) Then c_n is set equal to minus RMIN.

e) The right hand side value, h_m, is entered as:

$$h_m = k(FF)$$

where k is a number larger than the sum of the x_j in the optimal tableau.

The problem is now modified so it may be handled by use of the primal-dual algorithm.

The remainder of the program is a FORTRAN representation of the steps outlined in the sections "The Standard Primal-Dual Algorithm" and "Description of the New Technique." However, for computer calculations, an essential addition to the procedures is some control over round-off error.

Round-off error develops when two large numbers, say y and z, (differing only slightly from each other), are subtracted from each other such as:

$$x = y - z$$

In this program the absolute value of x divided by y is compared to EP. If it is less than EP, x is set equal to zero. As a further control, the values of DB, B, and H are updated each iteration.

BIBLIOGRAPHY

Cutler, Leola. "Tape for SHARE Linear Programming Test Problems" (Santa Monica, California: The RAND Corporation, 1962).

Dantzig, George B. Linear Programming and Extensions. Princeton: Princeton University Press, 1963.

Gass, Saul I. Linear Programming Methods and Applications. New York: McGraw-Hill Book Company, Inc., 1964.

Greenberg, Harold. "Modification of the Primal-Dual Algorithm for Degenerate Problems" (unpublished paper read at the Naval Postgraduate School, 1968).

Hadley, G. Linear Programming. Palo Alto: Addison-Wesley Publishing Company, Inc., 1962.

SHARE Linear Programming Committee. "SHARE Standard for Linear Programming Input Matrix Data Format" (Santa Monica, California: The RAND Corporation, 1960).

Shudde, Rex. H. "Primal-Dual Algorithm" (mimeographed class notes for use at the Naval Postgraduate School, 1967).

Wolfe, Philip and Leola Cutler. "Experiments in Linear Programming," Recent Advances in Mathematical Programming, eds. Robert L. Graves and Philip Wolfe. New York: McGraw-Hill Book Company, Inc., 1963.

APPENDIX

```
C         PROGRAM TO SOLVE PRIMAL-DUAL LINEAR
C         PROGRAMMING PROBLEMS
C
          IMPLICIT REAL*8 (A-H,O-Z)
          COMMON A(70,150),B(70,70),C(150),D(150),H(70),AX(70)
          COMMON DB(70),DUAL(70),R(150),CE(150),E,W,Z,DI,FF,EP
          COMMON CNAME(150)
          COMMON IB(70),M,N,JS,IR,INDEX,ITR,M1,N1
          REAL*8 NINE/'9999      '/
          FF=10000.
          EP=0.000001
          DI=1.
          M=0
          DO 1 I=1,70
          H(I)=0.
          DO 1 J=1,150
        1 A(I,J)=0.
          DO 2 I=1,150
          CNAME(I)=0.
          CE(I)=0.
        2 C(I)=0.
C
C         READING IN COEFICIENTS OF CONSTRAINT EQUATIONS
C
          DO 5 N=1,1000
          READ (5,3) CB, J, RB
          IF(CB.EQ.NINE) GO TO 16
        3 FORMAT (6X, A6, I6, F10.4)
          DO 4 I=1,150
          IF(CNAME(I).EQ.CB) GO TO 30
          IF(CNAME(I).EQ.0.) GO TO 30
          GO TO 4
       30 A(J,I)=RB*FF
          CNAME(I)=CB
          WRITE (6,102) I, J, A(J,I)
      102 FORMAT (1X, 2I10, F18.9)
          IF(J.LE.M) GO TO 5
          M=J
          GO TO 5
        4 CONTINUE
        5 CONTINUE
C
C         READING IN RIGHT HAND SIDE VALUES
C
       16 DO 6 N=1,100
          READ (5,14) I, RB
       14 FORMAT (I18, F10.4)
          IF(I.GT.999) GO TO 17
          H(I)=RB*FF
          WRITE (6,14) I, H(I)
        6 CONTINUE
C
C         READING IN COSTS
C
       17 DO 9 J=1,200
          READ (5,7) CC, II, RB
        7 FORMAT (6X, A6, I6, F10.4)
          IF(CC.EQ.NINE) GO TO 18
          DO 11 K=1,150
          IF(CC.EQ.CNAME(K)) GO TO 12
          GO TO 11
       12 C(K)=RB*FF
      CE(K)=RB
      WRITE (6,13) K, II, C(K)
   13 FORMAT (6X, 2I6, F15.5)
      GO TO 9
   11 CONTINUE
    9 CONTINUE
   18 CONTINUE
      DO 20 I=1,150
      IF(CNAME(I).GT.0.) GO TO 20
      N=I-1
      GO TO 21
   20 CONTINUE
   21 WRITE (6,101) M, N
  101 FORMAT (10X, 'NUMBER OF ROWS=', I5, 15X,
     1'NUMBER OF COLUMNS=', I5)
      WRITE (6,103) (CNAME(I), I=1,175)
  103 FORMAT (1X, 20A6)
      WRITE (6,104) (C(J), J=1,N)
  104 FORMAT (1X, 10E12.4)
      INDEX=1
      N1=N+1
      M1=M+1
      CALL LIP
    8 STOP
      END
      SUBROUTINE LIP
C
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON A(70,150),B(70,70),C(150),D(150),H(70),AX(70)
      COMMON DB(70),DUAL(70),R(150),CE(150),E,W,Z,DI,FF,EP
      COMMON CNAME(150)
      COMMON IB(70),M,N,JS,IR,INDEX,ITR,M1,N1
      E=0.5/DI
C
C     MODIFYING THE PROBLEM IF THE OBJECTIVE
C     FUNCTION CONTAINS NEGATIVE COSTS
C
      RMIN=0.
      DO 50 I=1,N
      IF(C(I).GE.E) GO TO 50
      IF(C(I).GE.RMIN) GO TO 50
      RMIN=C(I)
   50 CONTINUE
      IF(RMIN.EQ.0.) GO TO 60
      DO 52 I=1,N
   52 C(I)=C(I)-RMIN
      N=N+1
      M=M+1
      H(M)=1000.*FF
      C(N)=-RMIN
      DO 53 J=1,N
   53 A(M,J)=FF
C
C     INITIALIZING THE TABLEAU
C
   60 DO 3 J=1,M
      DO 2 I=1,M
    2 B(I,J)=0.
    3 B(J,J)=1.
      DO 4 I=1,M
      DB(I)=-1.
      DUAL(I)=0.
    4 IB(I)=-I
      DO 5 J=1,N
      R(J)=0.
      D(J)=0.
      DO 5 I=1,M
      D(J)=D(J)-A(I,J)
    5 CONTINUE
      W=0.
      DO 6 I=1,M
    6 W=W+H(I)
      ITR=0
C
C     OBTAINING A COST EQUAL TO ZERO AND A PIVOT COLUMN
C
    7 IF(DABS(W)-E)100,100,8
    8 DO 9 J=1,N
      IF(D(J))10,9,9
    9 CONTINUE
      GO TO 101
   10 JS=0
      DO 11 J=1,N
      IF(R(J).EQ.1.) GO TO 11
      IF(D(J).GE.-E) GO TO 11
      IF(JS.EQ.0) GO TO 13
   12 IF(CX+C(J)/D(J))11,11,13
   13 CX=-C(J)/D(J)
      JS=J
   11 CONTINUE
   47 DO 14 J=1,N
   14 C(J)=C(J)+CX*D(J)
      C(JS)=0.
      DO 301 J=1,N
      IF(C(J).LT.E) GO TO 301
      GO TO 302
  301 CONTINUE
      GO TO 100
  302 Z=0.
      DO 303 I=1,M
      J=IB(I)
      IF(J.LE.0) GO TO 303
      Z=Z+CE(J)*H(I)
  303 CONTINUE
      DO 40 I=1,M
   40 DUAL(I)=DUAL(I)-CX*DB(I)
   15 IF(W-E)100,100,41
   41 DO 16 J=1,N
      IF(C(J)-0.00001)17,17,16
   17 C(J)=0.
   18 IF(D(J)-D(JS))19,16,16
   19 JS=J
   16 CONTINUE
      WRITE (6,213) JS, D(JS)
  213 FORMAT (1H, 'JS=', I4, 3X, 'D=', F12.4)
      IF(D(JS))20,7,7
   20 CALL ELEM
      IF(DI-1.)528,26,26
   26 CALL PIVOT
      GO TO 15
C
C     PRINTING OF RESULTS
C
  100 WRITE (6,204)
  204 FORMAT(1H , 'RESULT')
      GO TO 425
  101 WRITE (6,205)
  205 FORMAT(1H , 'INFEASIBLE')
      STOP
  425 DO 426 I=1,M
  426 WRITE (6,526) IB(I), H(I), DUAL(I)
  526 FORMAT (1H , 'BASIS VARIABLE NUMBER=', I4, 5X,
     1'VALUE OF BASIS VARIABLE=', F12.4, 5X,
     1'DUAL=', F12.4)
      IF(RMIN.EQ.0.) GO TO 427
      ZZ=0.
      DO 429 K=1,M
      J=IB(K)
      IF(J.LE.0) GO TO 429
      ZZ=ZZ+CE(J)*H(K)
  429 CONTINUE
      GO TO 428
  427 ZZ=Z
  428 WRITE (6,527) W, ZZ
  527 FORMAT (1H , 'W=', F12.8, 5X, 'Z=', F12.5)
      IF(INDEX.EQ.1) GO TO 1
  528 RETURN
```

```
C
C     THE NEW TECHNIQUE
C
    1 IT=0
      DO 30 I=1,M
      IF(H(I).GT.-E) GO TO 30
      IF(IT.EQ.0) GO TO 31
      IF(H(I).GE.H(IT)) GO TO 30
   31 IT=I
   30 CONTINUE
      IF(IT.EQ.0) GO TO 528
      IB(IT)=-IT
      H(IT)=-H(IT)
      DO 34 K=1,M
      DB(K)=B(IT,K)
   34 B(IT,K)=-B(IT,K)
      DO 33 I=1,M
      IF(H(I).GT.-E) GO TO 33
      H(I)=H(I)+H(IT)
      DO 35 K=1,M
   35 B(I,K)=B(I,K)+B(IT,K)
   33 CONTINUE
      W=H(IT)
      DO 36 J=1,N
      AX(IT)=0.
      DO 21 K=1,M
   21 AX(IT)=AX(IT)+B(IT,K)*A(K,J)
   36 D(J)=-AX(IT)
      INDEX=0
      GO TO 8
      END
      SUBROUTINE ELEM
C
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON A(70,150),B(70,70),C(150),D(150),H(70),AX(70)
      COMMON DB(70),DUAL(70),R(150),CE(150),E,W,Z,DI,FF,EP
      COMMON CNAME(150)
      COMMON IB(70),M,N,JS,IR,INDEX,ITR,M1,N1
      IR=0
      J=JS
C
C     CALCULATING THE REVISED SIMPLEX TABLEAU PIVOT COLUMN
C     WITH ROUND-OFF ERROR CONTROL
C
      DO 22 I=1,M
      DM=0.
      DP=0.
      AX(I)=0.
      DO 21 K=1,M
      T=B(I,K)*A(K,J)
      IF(T.GE.0.) GO TO 40
      DM=DM+T
      GO TO 21
   40 DP=DP+T
   21 CONTINUE
      AX(I)=DP+DM
      IF(DP.EQ.0.) GO TO 22
      IF(DABS(AX(I)/DP).GT.EP) GO TO 22
      AX(I)=0.
   22 CONTINUE
C
C     FINDING THE PIVOT ROW
C
      JR=0
      EE=E
   31 DO 25 I=1,M
      IF(INDEX.EQ.0) GO TO 5
      IF(IB(I)) 5,25,25
    5 IF(AX(I)-E)25,25,23
   23 IF(IR.EQ.0) GO TO 24
      IF(DABS(H(I)-AX(I)*X)-EE)70,70,2
    2 IF(H(I)-AX(I)*X)24,70,25
   24 X=H(I)/AX(I)
    3 IR=I
      EE=E/AX(IR)
      GO TO 25
   70 IF(IB(I))74,25,73
   73 IF(IB(IR))25,25,4
   74 IF(IB(IR))4,25,3
    4 IF(AX(IR)-AX(I))25,25,3
   25 CONTINUE
      RETURN
      END
      SUBROUTINE PIVOT
C
C
      IMPLICIT REAL*8 (A-H,O-Z)
      COMMON A(70,150),B(70,70),C(150),D(150),H(70),AX(70)
      COMMON DB(70),DUAL(70),R(150),CE(150),E,W,Z,DI,FF,EP
      COMMON CNAME(150)
      COMMON IB(70),M,N,JS,IR,INDEX,ITR,M1,N1
      DIMENSION P(200),BX(200)
      WRITE (6,214) IR, AX(IR)
  214 FORMAT (1H, 25X, 'IR=', I3, 3X, 'AX=', F12.4)
C
C     PERFORMANCE OF THE PIVOT
C
   43 IF(IB(IR)) 1,1,2
    2 K=IB(IR)
      R(K)=0.
      GO TO 3
    1 P(JS)=1.
    3 IB(IR)=JS
      DI=DI*DABS(AX(IR))
      IF(DI.GE.9999998.) GO TO 40
      DI=IDINT(DI+0.5)
   40 CONTINUE
      E=0.5/DI
      DO 27 I=1,M
   27 P(I)=-AX(I)/AX(IR)
      P(IR)=1./AX(IR)-1.
      P(M+1)=-D(JS)/AX(IR)
      DO 28 J=1,M
      BX(J)=B(IR,J)
   28 CONTINUE
      HX=H(IR)
      DO 29 J=1,M
      DO 29 I=1,M
      T=B(I,J)
      B(I,J)=B(I,J)+P(I)*BX(J)
      IF(T.EQ.0.) GO TO 29
      IF(DABS(B(I,J)/T).GT.EP) GO TO 29
      B(I,J)=0.
   29 CONTINUE
      DO 30 I=1,M
      T=H(I)
      H(I)=H(I)+P(I)*HX
      IF(T.EQ.0.) GO TO 30
      IF(DABS(H(I)/T).GT.EP) GO TO 30
      H(I)=0.
   30 CONTINUE
      W=W-P(M+1)*HX
      DO 31 J=1,M
      T=DB(J)
      DB(J)=DB(J)+P(M+1)*BX(J)
      IF(T.EQ.0.) GO TO 31
      IF(DABS(DB(J)/T).GT.EP) GO TO 31
      DB(J)=0.
   31 CONTINUE
C
C     UPDATING THE DB, B, AND H VALUES
C
      IF(DI.GE.999998.) GO TO 81
      DO 92 J=1,M
      DF=DI*DABS(DB(J))+0.5
      IF(DF.GT.999998.) GO TO 61
      DF=IDINT(DF)/DI
      DB(J)=DSIGN(DF,DB(J))
   61 DO 92 I=1,M
      DF=DI*DABS(B(I,J))+0.5
      IF(DF.GT.999998.) GO TO 92
      DF=IDINT(DF)/DI
      B(I,J)=DSIGN(DF,B(I,J))
   92 CONTINUE
      DO 93 I=1,M
      DF=DI*DABS(H(I))+0.5
      IF(DF.GT.999998.) GO TO 93
      DF=IDINT(DF)/DI
      H(I)=DSIGN(DF,H(I))
   93 CONTINUE
   81 CONTINUE
      DF=W*DI
      IF(DF.GE.9999998.) GO TO 87
      W=IDINT(DI*W+0.5)/DI
   87 CONTINUE
C
C     CALCULATION OF D(J)'S FOR NEXT ITERATION
C
      DO 34 J=1,N
      DM=0.
      DP=0.
      D(J)=0.
      DO 33 K=1,M
      T=DB(K)*A(K,J)
      IF(T.GE.0.) GO TO 41
      DM=DM+T
      GO TO 33
   41 DP=DP+T
   33 CONTINUE
      D(J)=DP+DM
      IF(DP.EQ.0.) GO TO 34
      IF(DABS(D(J)/DP).LE.EP) GO TO 35
      IF(DABS(D(J))-E)35,35,34
   35 D(J)=0.
   34 CONTINUE
      D(JS)=0.
      IF(W.GT.0.001) GO TO 44
      W=0.
   44 ITR=ITR+1
      WRITE (6,210) ITR, W, Z, DI
  210 FORMAT (1H , 5X, 'ITR=', I5, 5X, 'W=', F18.8,
     15X, 'Z=', F12.4, 5X, 'DI=', E12.4)
   20 RETURN
   21 STOP
      END
```

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	20
2.	Library Naval Postgraduate School Monterey, California 93940	2
3.	Director, Systems Analysis Division (Op. 96) Office of the Chief of Naval Operations Washington, D. C. 20350	1
4.	Department of the Army Civil Schools Branch, OPO, OPD Washington, D. C. 20315	1
5.	Department of Operations Analysis Naval Postgraduate School Monterey, California 93940	1
6.	Professor Harold Greenberg Department of Operations Analysis Naval Postgraduate School Monterey, California 93940	1
7.	Captain John Edmund Easterbrook Box 1637, Naval Postgraduate School Monterey, California 93940	1

UNCLASSIFIED
Security Classification

DOCUMENT CONTROL DATA - R&D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)* Naval Postgraduate School Monterey, California 93940	2a. REPORT SECURITY CLASSIFICATION Unclassified
	2b. GROUP

3. REPORT TITLE
A Technique for Speeding Convergence in Solving Linear Programs

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*
Master of Science Thesis, June 1968

5. AUTHOR(S) *(Last name, first name, initial)*
Easterbrook, John E.

6. REPORT DATE June 1968	7a. TOTAL NO. OF PAGES 28	7b. NO. OF REFS 8
8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)	
b. PROJECT NO.		
c.	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*	
d.		

10. AVAILABILITY/LIMITATION NOTICES
[redacted].

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY

13. ABSTRACT

A technique for reducing the number of iterations necessary for solving linear programs using the primal-dual algorithm is presented. It appears that the new method will also decrease the number of iterations over any other simplex algorithm. A FORTRAN program incorporating the technique, as well as some comparative computational results are given.

DD FORM 1 JAN 64 1473

UNCLASSIFIED
Security Classification

UNCLASSIFIED
Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Linear Programming Primal-Dual Algorithm Modified Primal-Dual Computer Program for Primal-Dual						

DD FORM 1 NOV 65 1473 (BACK)
S/N 0101-807-6821

UNCLASSIFIED
Security Classification

A-31409